Set-up procedure to install and de-install software products

Patent number:	DE19924610
Publication date:	2001-01-11
Inventor:	DIESNER MARTIN (DE); JANSEN STEFAN (DE)
Applicant:	FUJITSU SIEMENS COMPUTERS GMBH (DE)
Classification:	
- international:	G06F9/445
- european:	G06F9/445N
Application number:	DE19991024610 19990528
Priority number(s):	DE19991024610 19990528

Report a data error here

Abstract of **DE19924610**

The procedure is carried out by modules which are called and processed one after the other. Common components are used as stand-alone modules in the set-up procedure. The common component contains information on how often it is called by other set-up programs to install or de-install. The common component is automatically de-installed if it is called by the last product installation to de-install. The modules are arranged in a hierarchy with a top module having a graphic user interface and sub-modules without graphic user interfaces. The modules are processed according to the hierarchy.

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENT- UND MARKENAMT

(12) **Offenlegungsschrift**

(10) **DE 199 24 610 A 1**

(51) Int. Cl.[7]: G 06 F 9/445

(21) Aktenzeichen: 199 24 610.6
(22) Anmeldetag: 28. 5. 1999
(43) Offenlegungstag: 11. 1. 2001

(71) Anmelder:
Fujitsu Siemens Computers GmbH, 81739 München, DE

(74) Vertreter:
Epping, W., Dipl.-Ing. Dr.-Ing., Pat.-Anw., 80339 München

(72) Erfinder:
Dießner, Martin, Dipl.-Inform. (FH), 86830 Schwabmünchen, DE; Jansen, Stefan, Dipl.-Ing. (FH), 86179 Augsburg, DE

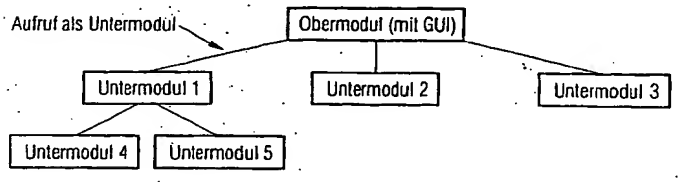
(56) Entgegenhaltungen:
US 57 21 824 A
RÖßMANN, M.: Applikatioen entwickeln unter Windows
NT 4.0, Addison-Wesley, 1997;

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

Prüfungsantrag gem. § 44 PatG ist gestellt

(54) Setup-Verfahren

(57) Es wird ein Setup-Verfahren zum Installieren und Deinstallieren von Software-Produkten oder -produktfamilien bereitgestellt, die wenigstens eine Komponente gemeinsam nutzen, wobei die gemeinsame Komponente bei der Installation der Produkte installiert oder einem Update unterworfen und bei der Deinstallation deinstalliert wird, wenn sie nicht mehr benötigt wird. Das Setup-Verfahren wird durch Module ausgeführt, die nacheinander aufgerufen und abgearbeitet werden. Die gemeinsamen Komponenten werden als eigenständige Module innerhalb des Setup-Verfahrens ausgeführt, und dazu wird ein eigenes Installationsprogramm und Deinstallationsprogramm ausgeführt.



DE 199 24 610 A 1

Beschreibung

Die Erfindung betrifft ein Setup-Verfahren zum Installieren und Deinstallieren von Software-Produkten oder -produktfamilien, die wenigstens eine Komponente gemeinsam nutzen, wobei die gemeinsame Komponente bei der Installation der Produkte installiert oder einem Upgrade unterworfen und bei der Deinstallation deinstalliert wird, wenn sie nicht mehr benötigt wird.

Bei einer Installation ohne Benutzerinteraktion werden generell zwei Arten der Installation unterschieden. Die Unattended-Installation ermöglicht die Installation ohne Benutzereingaben, es werden jedoch am Bildschirm verschiedene Meldungen angezeigt, die vom Benutzer wahrgenommen werden. Bei der Silent-Installation wird die Installation so durchgeführt, daß vom Benutzer keine Meldungen wahrgenommen werden können. Bei beiden Typen der Installation ist die Reaktionszeit des Computers heruntergesetzt und die Lampe für die Festplattentätigkeit leuchtet in unregelmäßigen Abständen.

Bei der Installation verschiedener Produkte einer Produktfamilie gibt es immer wieder Abhängigkeiten zwischen den einzelnen Produkten. Wird ein Produkt einer Produktfamilie auf einen Rechner installiert, dann werden unter Umständen auch Komponenten installiert, die von mehreren Produkten dieser Produktfamilie oder von Produkten einer anderen Produktfamilie gemeinsam verwendet werden (gemeinsame Komponente). Je nach Benutzerauswahl in den Produktinstallationen werden mehrere unterschiedliche gemeinsame Komponenten unterschiedlich häufig benötigt.

Das Problem stellt die Installation bzw. die Deinstallation der gemeinsamen Komponenten dar. Eine gemeinsame Komponente wird von dem einen oder anderen Produkt je nach Benutzerauswahl entweder installiert oder nicht. Wird eine gemeinsame Komponente von mehreren Produkten installiert, darf diese bei der Deinstallation eines dieser Produkte nicht deinstalliert, das heißt vom Rechner entfernt werden, sondern muß für die anderen Produkte noch zur Verfügung stehen. Erst bei der Deinstallation des letzten Produktes, das diese gemeinsame Komponente verwendet, darf und sollte diese deinstalliert werden.

Verschärft wird die Problematik dadurch, daß von verschiedenen Produkten unterschiedliche Versionen der gemeinsamen Komponente installiert werden können und entsprechend deinstalliert werden müssen. Produkt A installiert beispielsweise eine Version einer gemeinsamen Komponente. Diese wird durch die Installation von einem beliebigen Produkt H mit einer neueren Version der gemeinsamen Komponente ersetzt. Soll nun Produkt B wieder deinstalliert werden, dann muß die gemeinsame Komponente der beiden Produkte aber noch für Produkt A zurückblieben. Wird nun Produkt A deinstalliert, dann besteht das Problem, daß das Deinstallationsprogramm die neuere Version der gemeinsamen Komponente unter Umständen nicht in vollem Umfang kennt und somit Teile dieser gemeinsamen Komponente nicht deinstalliert. Dies führt dazu, daß auf dem Rechner eine Teilinstallation der gemeinsamen Komponente zurückbleibt. Mögliche Folgen dieser unvollständigen Deinstallation reichen von unnötig belegtem Speicherplatz auf der Festplatte bis hin zur Unbrauchbarkeit des Systems.

Erst zur Laufzeit der Installation wird vom Benutzer ausgewählt, welche Komponenten installiert werden müssen. Zudem hat der Benutzer die Möglichkeit die Produkte in unterschiedlicher Reihenfolge zu installieren und zu deinstallieren. Die Gesamtheit der Kombinationsmöglichkeiten führt bei der Freigabe der Produkte zu immer komplexeren Tests. Ein kompletter Test ist nicht mehr möglich und das Verhalten beim Benutzer damit nicht mehr vorhersehbar.

Zusätzliche Probleme entstehen dadurch, daß die Installationsprogramme der verschiedenen Produkte mit unterschiedlichen Installationsmethoden und Installationswerkzeugen erstellt werden, die nicht miteinander vereinbar sind. Die unterschiedlichen Installationswerkzeuge (Wise, InstallShield, NT-Setup, . . .) weisen bei der Verwaltung von gemeinsam benützten Komponenten unterschiedliche und teilweise erhebliche Schwächen auf. Durch verschiedene Anpassungen wurde versucht diese Probleme und Schwächen zu beheben. Ein einfacher Umstieg auf evtl. neuere und bessere Installationswerkzeuge ist dadurch aber nicht möglich.

Bisher wurden bei jeder Installation der einzelnen Produkte die verschiedenen gemeinsamen Komponenten unter Umständen mit verschiedenen Versionen installiert. Mit sehr großem Aufwand mußte darauf geachtet werden, daß die unterschiedlichen Produktinstallationen in Bezug auf die gemeinsame Komponente zueinander kompatibel waren. Es mußte auch darauf geachtet werden, daß nicht zu viele verschiedene Versionen der gemeinsamen Komponente in den einzelnen Produktinstallationen vorhanden waren. Wurde der Teil eines Installationsprogrammes eines Produktes verändert, der eine gemeinsame Komponente betraf, mußte der selbe Teil auch in den anderen Produkten entsprechend geändert werden. Damit wurde versucht, die Zahl der Versionen der gemeinsamen Komponente möglichst niedrig zu halten, bedeutete aber andererseits ständigen mehrfachen Entwicklungsaufwand. Auch passierte es immer wieder, daß durch unterschiedliche Freigabezeitpunkte der einzelnen Produkte mehrere unterschiedliche Versionen beim Kunden auftraten. Ein Konflikt zwischen den verschiedenen Versionen war oft nicht auszuschließen. Wie bereits oben beschrieben war der dafür notwendige Testaufwand sehr hoch.

Der Erfindung liegt die Aufgabe zugrunde, ein Setup-Verfahren für ein oder mehrere Installationsprogramme bereitzustellen, welches die verschiedenen vorhandenen Installationsprogramme vereinheitlicht und die Implementierung vereinfacht und bei dem insbesondere die Durchführung eines Installations-, Deinstallations und Upgradevorgangs ohne Benutzereingabe erfolgen kann.

Zur Lösung dieser Aufgabe ist das erfindungsgemäße Setup-Verfahren dadurch gekennzeichnet, daß das Setup-Verfahren durch Module ausgeführt wird, die nacheinander aufgerufen und abgearbeitet werden, und daß die gemeinsamen Komponenten als eigenständige Module innerhalb des Setup-Verfahrens ausgeführt werden und dazu ein eigenes Installationsprogramm und Deinstallationsprogramm ausführen.

Durch das erfindungsgemäße Verfahren wird ein modulares Installationskonzept bereitgestellt, welches gemeinsame Komponenten in eigenständige Installationsprogramme auslagert. Wird nun das Installationsprogramm der gemeinsamen Komponente geändert, betrifft dies nicht mehr das Installationsprogramm des gesamten Produktes. Durch Änderungen kann es nun weiterhin zu unterschiedlichen Versionen der gemeinsamen Komponente kommen. Da diese gemeinsame Komponente ein eigenes Installationsprogramm und ein eigenes Deinstallationsprogramm besitzt, kann die gemeinsame Komponente vollständig vom Rechner entfernt werden und es verbleiben keine Teilinstallationen mit den

oben beschriebenen Nachteilen. Werden Änderungen am Installationsprogramm durchgeführt, dann kann auf einfache Weise das geänderte Modul ausgetauscht werden. Durchzuführende Tests müssen nicht so umfangreich gemacht werden, da nur die einzelnen Module und deren Zusammenspiel getestet werden muß.

Die unterschiedlichen Abhängigkeiten die dadurch entstanden sind, daß je nach Benutzerauswahl mehrere gemeinsame Komponenten installiert werden müssen, sind nun dadurch gelöst, daß eine gemeinsame Komponente von der eigentlichen Produktinstallation gezielt installiert und deinstalliert wird.

Bei dem erfindungsgemäßen Verfahren ist es möglich, sowohl die eigentliche Produktinstallation, als auch die damit installierten Komponenten als Modul zu implementieren. Somit werden die Produktinstallationen als auch die gemeinsamen Komponenten zu gleichwertigen Modulen. Sie unterscheiden sich nur dadurch, daß die Produktinstallation zu einem Modul mit Benutzeroberfläche wird, während eine Komponente als Modul ohne Benutzeroberfläche aufgerufen wird. Im Allgemeinen wird das Modul ohne Benutzeroberfläche (Untermodul) immer von einem Modul mit Benutzeroberfläche (Obermodul) aufgerufen. Zwischen den einzelnen Modulen muß dazu während der Installation kommuniziert werden. Wenn die Art der Kommunikation festgelegt ist, dann können die Module aus verschiedenen ausführbaren Programmen bestehen.

Ein Installationsprogramm kann beispielsweise mit Wise, InstallShield, C++ oder dergleichen erstellt werden. Muß ein Modul geändert werden, dann kann es auch in einer anderen "Programmiersprache" entwickelt werden und kann völlig transparent wieder eingefügt werden, da die Funktionalität nach außen gleichbleibt.

Durch die Art der Implementierung wird auch die Möglichkeit geschaffen, daß ein Modul sowohl mit Benutzeroberfläche als auch ohne installiert werden kann. Damit ist es für einen Administrator möglich, das Installationsprogramm auf einem Computer im Netzwerk zur Verfügung zu stellen und durch entsprechende Maßnahmen (beispielsweise Loginskript) auf den Rechnern (Clients) zu installieren, ohne daß der Benutzer dies merkt.

Fehlende oder nicht benötigte Produktinstallationen oder Teilkomponenten können ohne besonderen Aufwand weggelassen oder hinzugefügt werden.

Eine vorteilhafte Ausgestaltung des erfindungsgemäßen Verfahrens ist dadurch gekennzeichnet, daß in der gemeinsamen Komponenten festgestellt und gespeichert wird, wie oft sie von anderen Setup-Programmen zur Installation bzw. Deinstallation aufgerufen wurde, und daß die gemeinsame Komponente dann automatisch deinstalliert wird, wenn sie von der letzten Produktinstallation zur Deinstallation aufgerufen wird, die die gemeinsame Komponente noch benutzt hat.

Jede Komponente ist im Zuge der Modularisierung so entwickelt, daß sie selbständig erkennt, wie oft sie von unterschiedlichen Installationsprogrammen installiert, das heißt aufgerufen wurde, kann sie auch erkennen, wann die eigentliche Deinstallation der Komponente durchgeführt werden muß. Eine Komponente deinstalliert sich genau dann, wenn sie von der letzten Produktinstallation zur Deinstallation aufgerufen wird, die diese Komponente noch benützt. Durch die Art der Implementierung ist sichergestellt, daß die Komponente auch noch zu früheren installierten Versionen kompatibel ist und diese entsprechend deinstalliert werden können.

Eine vorteilhafte Ausgestaltung des erfindungsgemäßen Verfahrens ist dadurch gekennzeichnet, daß die Module in eine Hierarchie aus wenigstens einem Obermodul mit Benutzeroberfläche und Untermodulen ohne Benutzeroberfläche eingegliedert werden, und daß die Module entsprechend der Hierarchie abgearbeitet werden. Damit wird erreicht, daß das Setup-Verfahren von der Installation eines Modules zur Installation des nächstens Moduls erst dann übergeht, wenn die Installation des vorhergehenden Moduls abgeschlossen ist, so daß es keine Überschneidungen gibt.

Eine weitere vorteilhafte Ausgestaltung des erfindungsgemäßen Verfahrens ist dadurch gekennzeichnet, daß beim Hinzufügen eines neuen Obermoduls ein bisheriges Obermodul als Untermodul aufgerufen wird, und daß vorzugsweise das neue Obermodul die gesamte Benutzeroberfläche beinhaltet.

Ein bisheriges Obermodul mit Benutzeroberfläche kann somit auch als Untermodul aufgerufen werden. In diesem Fall muß ein neues Obermodul existieren, evtl. ein Modul zur Installation der gesamten Produktfamilie oder noch höher angesiedelt und nicht nur der einzelnen Produkte. Dieses neue Obermodul kann dann die komplette Benutzeroberfläche beinhalten. Es ruft die bisherigen Obermodule als Untermodul auf, die dann keine Benutzeroberfläche mehr anzeigen. Die Ober-/Untermodule können dann wie bisher die benötigten Untermodule aufrufen. Zusätzlich können vom neuen Obermodul auch die bisher existierenden Untermodule aufgerufen werden.

Man spricht von einem Obermodul, wenn es eine Benutzeroberfläche besitzt und als eigenständiges Installationsprogramm aufgerufen werden kann. Ein Obermodul generiert immer einen Uninstalistring in der Registry, damit es über die Systemsteuerung deinstalliert werden kann. Wird ein Obermodul als Untermodul (Parameter/Launcher in der Kommandozeile) aufgerufen, dann muß es dessen Funktionalität vollständig erfüllen. Wird ein Obermodul von einem weiteren Obermodul als Untermodul aufgerufen, dann handelt es sich bei dem neuen Obermodul meist um das Installationsprogramm für die gesamte Produktfamilie.

Eine weitere vorteilhaft Ausgestaltung des erfindungsgemäßen Verfahrens ist dadurch gekennzeichnet, daß die Hierarchie bzw. die Aufrufreihenfolge in einer Anweisungstabelle festgehalten wird, die vorzugsweise außerhalb der Module angelegt wird.

Durch eine optional außerhalb der Module liegende Anweisungstabelle können die Module erkennen, auf welcher Position einer völlig freien Modulhierarchie sie stehen. Wird die Anweisungstabelle geändert, dann werden die Module in anderen Aufrufreihenfolgen (andere Hierarchie) aufgerufen, was unter Umständen die Funktionalität eines Produktes oder der gesamten Produktfamilie entscheidend ändert. Über eine solche Modulhierarchie mit externer Anweisungstabelle können Produkte ohne wesentlichen Aufwand in verschiedene Produktfamilien integriert werde, auch in solche, in denen der Einsatz eines bestimmten Produktes bisher nicht vorgesehen war. Zudem können bisher bestehende Produktinstallationen neue zusätzliche Funktionen ohne Änderung des Installationsprogrammes installieren, wenn ein neues Modul hinzugefügt und die Anweisungstabelle entsprechend geändert wird.

Zur Kommunikation wird im Allgemeinen die Kommandozeile, eine INI-Datei, die Registry oder andere Möglichkeiten des Datenaustausches verwendet. Die Anweisungstabelle kann optional sein und ist im Allgemeinen als INI-Datei oder Registry-Tabelle oder jede andere Möglichkeit der Datenhaltung abgespeichert.

Die Installationsmodule sind für die Durchführung der Installation an sich verantwortlich. In diesen Modulen werden beispielsweise Dateien kopiert, Treiber installiert und Registry-Einträge vorgenommen. Ein Installationsmodul kann über eine Benutzeroberfläche verfügen, wenn Interaktionen mit dem Benutzer notwendig sind. Wird die Installation von einem Administrator über das Netzwerk durchgeführt, ohne daß der Benutzer dies merken soll, dann verwendet das Installationsprogramm automatisch eine Auftragsdatei, die beschreibt, wie zu installieren ist.

Eine weitere vorteilhafte Ausgestaltung des erfindungsgemäßen Setup-Verfahrens ist dadurch gekennzeichnet, daß beim Start einer Installation geprüft wird, ob ein Zählerstand in einem Modulcounter vorhanden ist, daß, wenn kein Zählerstand in dem Modulcounter vorhanden ist, und es sich damit um eine Neuinstallation handelt, der Zählerstand auf 1 gesetzt, ein Uninstallstring gesetzt und ein Display Name erzeugt wird, daß, wenn ein Zählerstand in dem Modulzähler vorhanden ist, ein Uninstallstring erzeugt und der Zählerstand des Modulcounters erhöht wird, wenn das Modul zum ersten Mal als Obermodul aufgerufen wurde, und daß dann die Installation der gemeinsamen Komponente durchgeführt wird. Hierbei handelt es sich um ein vorteilhaftes Verfahren, mit dem in dem Modul festgestellt werden kann, wie oft die gemeinsame Komponente installiert werden soll.

Eine weitere vorteilhafte Ausgestaltung des erfindungsgemäßen Setup-Verfahrens ist dadurch gekennzeichnet, daß bei der Installation einer gemeinsamen Komponente durch ein Modul, die in Form eines Untermoduls installiert wird, geprüft wird, ob das Untermodul (gemeinsame Komponente) bereits installiert war, daß, wenn der Untermodul bereits installiert war und kein Upgrade vorliegt, der Zählerstand des Modulcounters des Untermoduls erhöht, das Untermodul als Client eingetragen, die Deinstallationsroutine vermerkt und die Installation an dem Untermodul durchgeführt wird, und daß, wenn das Untermodul noch nicht installiert war, der Zählerstand des Modulcounters des Untermoduls um den Wert des eigenen Modulcounters erhöht, das Update Flag gelöscht, das Untermodul als Client eingetragen, die Deinstallationsroutine vermerkt und die Installation des Untermoduls durchgeführt wird. Auf diese Weise wird der Zählerstand in dem Untermodul immer auf den neuesten Stand gebracht, d. h. bei jeder Neuinstallation upgedatet.

Eine weitere vorteilhafte Ausgestaltung des erfindungsgemäßen Setup-Verfahrens ist dadurch gekennzeichnet, daß bei einer Deinstallation geprüft wird, ob andere Module aufgerufen werden sollen, daß, wenn andere Module aufgerufen werden sollen, die Module zur Deinstallation aufgerufen werden, daß, wenn im Client noch ein Zählerstand im Modulcounter vorhanden ist, ein weiteres Modul aufgerufen wird und daß, wenn im Client kein Zählerstand im Modulcounter vorhanden ist, der Clientvermerk gelöscht und die Deinstallationsroutine fortgesetzt wird. Mit dieser Verfahrensweise ist sichergestellt, daß die gemeinsame Komponente erst dann gelöscht wird, wenn sie von keinem Modul mehr gebraucht wird, aber auch nur dann.

Schließlich ist eine weitere vorteilhafte Ausgestaltung des erfindungsgemäßen Setup-Verfahrens dadurch gekennzeichnet, daß, wenn der Zählerstand im Modulcounter gleich "1" ist, geprüft wird, ob es einen Zählerstand in einem Shared-Counter gibt, daß, wenn ja eine Deinstallation durchgeführt wird, die Files gelöscht werden und der Shared-Counter erniedrigt wird, daß, wenn nein alle Dateien deinstalliert und alle Einträge gelöscht werden, und daß, wenn der Zählerstand im Modulcounter ungleich "1" ist, der Zählerstand im Modulcounter um "1" erniedrigt und der Modul deinstalliert wird.

Ausführungsbeispiele der Erfindung werden nun anhand der beiliegenden Zeichnungen beschrieben. Es zeigen:

Fig. 1 ein Blockdiagramm einer Hierarchie, bestehend aus einem Obermodul und mehreren Untermodulen;

Fig. 2 ein Blockdiagramm einer Hierarchie, bestehend aus zwei Obermodulen mit Benutzeroberfläche und mehreren Untermodulen;

Fig. 3 ein Ablaufdiagramm für die Installation eines Modules;

Fig. 4 ein Ablaufdiagramm für die Installation eines Untermoduls;

Fig. 5 ein Ablaufdiagramm einer Deinstallation eines Moduls;

Fig. 6 ein Beispiel einer hierarchischen Struktur des Setup-Verfahrens implementiert mit einem Wise-Installer; und

Fig. 7 ein Blockdiagramm für die Deinstallation bei dem Wise-Installer.

In **Fig.** 1 ist ein Obermodul mit Benutzeroberfläche (GUI = Graphical User Information). Entsprechend der Hierarchie werden von dem Obermodul der Untermodule **1** und von diesem die Untermodule **4** und **5** aufgerufen. Von dem Obermodul wird auch das Untermodul **2** und das Untermodul **3** entsprechend der Hierarchie aufgerufen.

In **Fig.** 2 ist die Variante gezeigt, bei der ein Obermodul als Untermodul aufgerufen, wobei es sich dann bei dem neuen Obermodul meist um das Installationsprogramm für die gesamte Produktfamilie handelt.

Man spricht von einem Untermodul, wenn ein Modul keine Benutzeroberfläche besitzt und nur im Silent-Mode (bei Wise mit Kommandozeilenparameter /S) aufgerufen werden kann. Ein Untermodul wird immer mit dem Parameter /Launcher in der Kommandozeile aufgerufen.

Die Anforderungen, die ein Modul hat, bestehen darin, daß das Installationsmodul wissen muß, ob ein Upgrade (Installation mit Beibehalten von gewissen Einstellungen der vorher installierten Version), eine Installation (komplett neue Installation) oder eine Deinstallation durchzuführen ist, ob die Installation ohne Benutzerinteraktion durchgeführt werden soll, welche Installationsoptionen zu wählen sind (Eingabe über Benutzeroberfläche oder Auftragsdatei), welche Komponenten, die in diesem Modul behandelt werden, zu installieren sind. Das Installationsprogramm muß ferner wissen, welche Untermodule zu installieren sind, und es führt Buch über den aktuellen Stand seines Ablaufs und ermöglicht somit das Wiederaufsetzen einer halbfertigen Installation.

Eine modulare Installation besteht üblicherweise aus einem oder mehreren Obermodulen, Untermodulen, INI Dateien, die für die Kommunikation zwischen Obermodulen und Untermodulen und zum Abwickeln der Installation ohne Benutzerinteraktion verantwortlich sind.

Das Obermodul führt gewöhnlich alle Benutzerabfragen durch, die nötig sind, um einen Installationsdurchlauf vollständig zu definieren. Das Installationsprogramm des Obermoduls kann dabei in verschiedenen Modi ablaufen. Im Administratormodus erfolgt ein Aufruf zum Schreiben einer Auftragsdatei, oder im Benutzermodus, bei dem eine Benutzeroberfläche vorhanden ist und der immer dann abläuft, wenn keine Auftragsdatei vorhanden ist, oder im Netzinstallationsmodus ohne Benutzeroberfläche immer dann, wenn die Auftragsdatei gefunden wird.

Im Administratormodus wird die Benutzeroberfläche der Installation dazu verwendet durch einen Administrator eine

Auftragsdatei für die Installationsmodule zu erstellen. Zusätzlich wird in diesem Modus erfragt, wie das Resultat der Installation festzuhalten ist (beispielsweise Ereignisanzeige unter NT, Pfad und Name für die Ergebnisdatei; SMS . . .). Wenn die Auftragsdatei erstellt ist, ist der Administratormodus beendet, das heißt es wird keine Installation im eigentlichen Sinne durchgeführt, sondern nur die Auftragsdatei geschrieben.

Im Benutzermodus wird das Programm vom Benutzer gestartet und alle Abfragen werden vom Benutzer beantwortet (Benutzeroberfläche) und daraufhin wird die eigentliche Installation durchgeführt.

Im Netzinstallationsmodus wird die Installation aufgerufen und verwendet automatisch eine im Administratormodus erstellte Auftragsdatei, die vorzugsweise im gleichen Verzeichnis liegt, in dem die Installation selber liegt. Dem Benutzer wird beispielsweise im Login Skript ein Start der Installation eingestellt. Die automatische Verwendung des Netzinstalltionsmodus beim Vorliegen der Auftragsdatei muß automatisch erfolgen. Ein Benutzer der die Installation von sich aus startet ist somit ebenfalls gezwungen, den vordefinierten Weg der Installation durchzuführen. Der Netzinstallationsmodus wird immer als Unattended-Installation oder Silent-Installation durchgeführt.

Technisch gesehen besteht ein Obermodul aus drei Hauptkomponenten, nämlich der graphischen Benutzeroberfläche (Was muß vom Benutzer/Administrator abgefragt werden?), der Starterkomponente (Welche Untermodule sind zu starten?)und dem Checker (sind alle Bedingungen für die Untermodule erfüllt?)

Das Obermodul liest Daten aus der Datei Modulbeschreibungdatei, schreibt im Benutzer- oder Netzinstallationsmodul seinen Fortgang in die Datei Statusdatei und erzeugt im Benutzer- oder Administratormodus die Datei Auftragsdatei. Das Ziel ist, daß das Obermodul aus den Einträgen in der Modulbschreibungdatei seine Auswahldialoge aufbauen und erkennen kann, welche Untermodule aufzurufen sind. Untermodule die eigene zusätzliche Abfragen benötigen, können dies in Form von Bibliotheken (DLL's) mit Benutzerdialogen tun, die in das Obermodul eingebunden werden.

Die graphische Benutzeroberfläche mit "silent" Option ermöglicht, daß alle Installationsoptionen vom Benutzer oder Administrator eingegeben werden können. Wird die Installation "normal" gestartet, dann kann der Benutzer verschiedene Optionen eingeben und die Installation durchführen. Wird die Installation im Administratormodus gestartet, dann wird nur die Benutzeroberfläche abgearbeitet und anschließend die Installation abgebrochen. Der Administrator kann dabei alle erforderlichen Optionen eingeben und eine Auftragsdatei erstellen.

Die Starterkomponente des Obermoduls ("Launcher") führt generell eine Installation und/oder eine Deinstallation durch. Bei der Deinstallation werden alle Untermodule aufgerufen, die nicht mehr benötigt werden und deshalb deinstalliert werden können. Bei der Installation werden alle Untermodule aufgerufen, die neu installiert werden müssen, oder nur upgedated werden sollen. Die Reihenfolge der Installation bzw. Deinstallation wird durch eine Modulpriorität in der Modulbeschreibungsdatei festgelegt.

Bei der Deinstallation und bei der Installation können mehrere Neustarts des Betriebssystems durchzuführen sein, bis die gesamte Installation vollständig abgeschlossen ist. Die Starterkomonente ist dann dafür verantwortlich, daß die Installation nach einem Neustart des Betriebssystems, der von einem Untermodul benötigt wird, wieder an geeigneter Stelle fortgesetzt werden kann.

Das Obermodul muß einen Wartemechanismus implementieren, der es ermöglicht ein Untermodul nach dem anderen zu installieren und jeweils zu warten, bis das Untermodul seine Installation abgeschlossen hat.

Der Checker überprüft, ob die benötigten Voraussetzungen für die zu installierenden Untermodule erfüllt sind, beispielsweise ob genügend Speicherplatz vorhanden ist. Der Checker kann die Installation abbrechen, wenn die Voraussetzungen nicht erfüllt sind. Wird mit Benutzerinteraktion gearbeitet, dann gibt der Checker eine entsprechende Fehlermeldung am Bildschirm aus. Wenn ohne Benutzerinteraktion gearbeitet wird, dann schreibt er in die Ergebnisdatei eine entsprechende Fehlermeldung, wenn dies vom Administrator gewünscht wird.

Ein Untermodul wird immer von einem Obermodul zur Installation oder Deinstallation mit dem Aufruf eines Untermoduls aufgerufen. Es besitzt keine Benutzeroberfläche und kann deshalb nur im Modus ohne Benutzeroberfläche aufgerufen werden. Wird ein Obermodul als Untermodul aufgerufen, dann darf die Benutzeroberfläche, die ein Obermodul gewöhnlich hat, nicht erscheinen und es müssen alle Einstellungen, die für die Installation wichtig sind aus der Auftragsdatei gelesen werden.

In der Modulbeschreibungsdatei werden die Anforderungen (Speicherplatz, Rechte, . . .) der Installationsmodule vom Moduldesigner, so wie die Abhängigkeiten der einzelnen Module voneinander festgehalten. In dieser Datei kann bei vollständig ausgereifter Implementierung auch die Aufrufhierarchie der einzelnen Module geregelt werden. Diese Datei darf während des Setups im schreibgeschützten Bereich liegen Die Auftragsdatei beinhaltet alle Informationen, die den Ablauf des gesamten Installationsmoduls steuern. Nicht besetzte Werte sind mit Standardwerten zu belegen. Der Pfad zur Auftragsdatei wird dem Setupmodul über Parameter mitgeteilt.

Die Statusdatei dient der Kommunikation zwischen Obermodulen und Untermodulen. Informationen die temporär vom Obermodul an ein oder mehrere Untermodule bzw. umgekehrt mitgeteilt werden müssen, können hier eingetragen werden. Diese Datei muß während des Ablaufs der Installation immer schreibbar sein und die Datei sollte im Windows-Temp-Verzeichnis abgelegt werden.

Folgende Einträge sind definiert

1. Reboot und Restart-Meldungen vom Untermodul

In der Datei Statusdatei wird unter anderem vom Untermodul protokolliert, ob das Obermodul einen Reboot oder Restart des Systems oder dergleichen für das Untermodul durchzuführen hat. Das Obermodul sollte immer darauf achten, den Neustart für mehrere Untermodule nur einmal zu machen.

Nachfolgend sind als Beispiele die Einträge für Reboot und Restart in der Statusdatei aufgeführt, wobei folgende Keys möglich sind.

Der Eintrag "Reboot = 0 | 1" wird von einem Modul gemacht. Damit erkennt das Obermodul, daß es einen Reboot durchführen muß, damit ein Untermodul fertig installiert/deinstalliert werden kann. Wird ein Reboot durchgeführt, ist ein Restart überflüssig.

Der Eintrag "Restart = 0 1 1" wird von einem Modul gemacht. Damit erkennt das Obermodul, daß es einen Restart von Windows durchführen muß, damit ein Untermodul fertig installiert/deinstalliert werden kann.

2. SetupID des Obermoduls

Der Eintrag "Sektion = [SetupID]" teilt den Untermodulen mit, von welchem Obersetup sie aufgerufen wurden. Stimmt diese ID mit der ID des Moduls in der Registry überein wird das Modul zur Beschleunigung des Installationsvorganges nicht noch einmal neu installiert. Als Key ist möglich "Date Time = Aktuelles Datum und Zeit".

Ergebnisdatei

Der Administrator gibt in der Auftragsdatei an, ob und wo die Installationsprogramme bei der Installation ohne Benutzerinteraktion ihre Ergebnisdateien ablegen sollen. Der Administrator gibt dabei nur den Pfad der Ergebnisdatei an. Der Dateiname selbst ist immer der Computername des Rechners, <Computername>.ini. Wird vom Administrator ein Netzlaufwerk angegeben, dann befindet sich nach der Installation von jedem Computer eine INI-Datei in dem angegebenen Verzeichnis. Über den Explorer kann dann nach dem Inhalt der Dateien gesucht werden. Wird beispielsweise nach "Error" gesucht, dann werden alle Dateien aufgelistet, in denen ein Fehler enthalten ist und damit alle Computer bei denen eine Installation fehlerhaft abgelaufen ist.

Von den Installationsprogrammen darf die Datei nicht bei jedem Schreiben neu angelegt werden, sondern nur ergänzt werden. Das heißt vorhandene Sektionen in der Ergebnisdatei müssen erhalten bleiben. Es müssen sowohl gemappte Pfade (f:\myRechner\respath <Computername>.ini) als auch UNC Notierungen (\\MYRechner\respath\<Computername>.ini) verwendet werden können.

In **Fig.** 3 ist ein Teil des Setup-Verfahrens als Flußdiagramm dargestellt, wobei **Fig.** 4 eine Fortsetzung des Flußdiagramms von **Fig.** 3 ist. Nach dem Start wird zunächst ausgeschlossen, daß ein Downgrade erfolgt.

Als nächstes wird festgestellt, ob ein Zählerstand in dem Modulcounter vorhanden ist (ist ein Modulcounter vorhanden?). Wenn ja, wird abgefragt, ob es sich um eine erste Installation handelt, und, wenn ja, wird eine Installation durchgeführt. Wenn festgestellt wird, daß kein Modulcounter vorhanden ist, wird der Modulcounter auf 1 gesetzt, und es wird ein Uninstallstring und ein Displayname erzeugt, wobei als nächstes wiederum die Frage nach einer Erstinstallation gestellt wird.

Wenn die Frage nach einer Erstinstallation mit nein beantwortet wird, wird festgestellt, ob ein Aufruf durch ein anderes Modul erfolgt. Wenn ja, wird gegebenenfalls ein Flag für den Sharedcounter beibehalten, was aber nur in dem noch beschriebenen Beispiel in Verbindung mit einer älteren Version von Unwise.exe notwendig ist, und die Installation wird dann durchgeführt. Wenn die Frage nach dem Aufruf eines anderen Moduls mit nein beantwortet wird, wird entschieden, ob ein Uninstallstring bereits gesetzt wurde, wenn ja, wird ein Upgradeflag gesetzt, u. U. ebenfalls ein Flag für den Sharedcounter beibehalten und eine Installation durchgeführt. Wenn nein, wird ein Uninstallstring erzeugt, der Modulcounter erhöht, u. U. ebenfalls ein Flag für den Sharedcounter beibehalten und die Installation durchgeführt. Nachdem die Dateien installiert sind, geht das Verfahren am Punkt a (**Fig.** 3 und 4) in das Verfahren von **Fig.** 4 über.

Wurde ein Modul bereits einmal installiert (keine Erstinstallation), dann wird modulintern immer ein Update gefahren, d. h. alle zu installierenden Dateien werden neu kopiert. Registry-Einträge, die einen Update überleben müssen, dürfen nicht neu gesetzt werden.

Das Aufrufen anderer Module (Untermodule, gemeinsame Komponenten) erfolgt gemäß **Fig.** 4. Es wird zunächst geprüft, ob andere Module aufgerufen werden sollen. Dies ist entweder fest vorgegeben durch eine starre Implementierung, oder das Modul erkennt dies aus einer Modulbeschreibungsdatei oder anhand eines bereits bestehenden Clienteintrages. Wenn ja, wird als nächstes überprüft, ob das Untermodul bereits durch das gerade ablaufende Modul installiert wurde, was durch einen Clienteintrag erkannt werden kann. Wenn nein, wird der Modulcounter des Untermoduls um den Wert des eigenen Modulcounters erhöht, und das Updateflag wird gelöscht. Wenn ja, wird festgestellt, ob ein Update durchgeführt werden soll, was durch das Updateflag erkannt wird. Dies wird beispielsweise beim Kommandozeilenparameter/Upgrade gesetzt.

Wenn die Frage nach dem Upgrade mit nein beantwortet wird, wird der Modulcounter des Untermoduls erhöht. Wenn die Frage nach dem Upgrade mit ja beantwortet wird, wird das Modul zum Upgrade aufgerufen und die entsprechende Installation durchgeführt. Nachdem der Modulcounter des Untermoduls auf den neuesten Stand gebracht wurde, wird das Untermodul als Client eingetragen und die Deinstallationsroutine vermerkt, worauf das Modul zur Installation aufgerufen wird. Am Ende dieser Routine wird wieder am Anfang weiterverfahren, wenn noch weitere Module aufgerufen werden müssen. Wenn alle Module aufgerufen sind, endet das Verfahren.

Fig. 5 zeigt das Ablaufdiagramm für die Deinstallation der Module. Zunächst wird festgestellt, ob andere Module aufgerufen werden sollen. Wenn ja, werden die Module zur Deinstallation aufgerufen und es wird entschieden, ob der Client (das Untermodul) noch einen Zählerstand im Modulcounter hat. Wenn ja, kehrt die Routine zum Ausgangspunkt zurück, wenn nein, wird der Clientvermerk des Untermoduls gelöscht und die Routine kehrt zum Ausgangspunkt zurück. Wenn kein weiterer Modul aufgerufen werden muß, wird festgestellt, ob der Modulcounter gleich "1" ist. Wenn nein, wird der Modulcounter um "1" erniedrigt, und bei der nächsten Frage wird entschieden, ob die Deinstallation über den Uninstallstring aufgerufen wurde. Wenn ja, wird der Uninstallstring vernichtet, und das Verfahren ist abgeschlossen, wenn nein, ist das Verfahren ebenfalls abgeschlossen.

Wenn festgestellt wird, daß der Modulcounter ungleich 1 ist, wird festgestellt, ob es einen Zählerstand in einem Sharedcounter gibt, und, wenn ja, wird festgestellt, ob der Sharedcounter der Datei (z. B. Exe) = 1 ist. Wenn nein, erfolgt eine normale Deinstallation, bei der die Dateien gelöscht und dadurch deren Sharedcounter erniedrigt wird. Danach wird das Verfahren vor der Entscheidung über die Deinstallation über den Uninstallstring fortgesetzt. Wenn bei der Entscheidung, ob es einen Sharedcounter gibt und ob der Sharedcounter der Datei (z. B. Exe) = 1 ist, mit nein bzw. ja beantwortet wird, werden die Dateien deinstalliert und alle Einträge vernichtet, und es darf kein Reboot für Inuse-Files erfolgen, und das

Verfahren wird vor der Entscheidung über die Deinstallation über den Uninstallstring fortgesetzt.

Dateien deinstallieren und alles putzen bedeutet, daß alle Einträge gelöscht werden, die jemals von diesem Modul erzeugt wurden, inklusive des Uninstallstrings. Sind Inuse-Dateien vorhanden, dann werden alle Einträge gelöscht, die zu diesem Inuse führten und die Dateien werden zur Delete-Liste des Betriebssystems hinzugefügt, damit diese nach einem Reboot gelöscht werden.

"Normale" Deinstallation bedeutet, daß so deinstalliert wird, wie es bisher in einem noch nicht modularen Verfahren üblich war. Der Sharedcounter wird erniedrigt, notwendige Dateien zur Kompatibilität mit alten Installationen können zurückbleiben. Sind Inuse-Dateien vorhanden, dann werden alle Einträge gelöscht, die zu diesem Inuse führten, und die Dateien werden zur Delete-Liste hinzugeführt, damit diese nach einem Reboot gelöscht werden. Damit kann ein Übergang von einem nicht modularen Verfahren auf das modulare Setup-Verfahren erfolgen.

Damit das Installationsmodul entscheiden kann, was zu tun ist, müssen folgende Kommandozeilenparameter ausgewertet werden:

"/S" gibt an, daß die Installation ohne Benutzerinteraktion im Netzinstallationsmodus durchgeführt werden soll. Die Auftragsdatei muß vorhanden sein.

"/Install" gibt an, daß das Installationsmodul installiert werden soll.

"/Remove" gibt an, daß das Installationsmodul deinstalliert werden soll.

"/Upgrade" gibt an, daß ein Upgrade durchgeführt wird. Werden weitere Untermodule aufgerufen, so müssen diese ebenfalls mit dem Schalter /Upgrade aufgerufen werden. Werden vom Installationsmodul keine Untermodule aufgerufen, dann muß dieser Parameter nicht ausgewertet werden.

"/Launcher" gibt an, daß das Installationsmodul vom einem anderen Installationsmodul aufgerufen wurde. Wird vom Installationsmodul ein Neustart benötigt (Neustarts sind nur am Ende der Installation möglich!), so ist in der Auftragsdatei das Flag "Reboot" oder das Flag "Restart" in der Sektion [SNINSTAL] auf 1 zu setzen. Der Aufrufer des Moduls hat für den abschließenden Neustart zu sorgen (evtl. für mehrere Module gemeinsam). Wurde ein Installationsmodul ohne diesen Parameter aufgerufen, so ist es das oberste Installationsmodul der Hierarchie.

"/Admin" gibt an, daß das Installationsmodul nur die Oberfläche anzeigen darf und alle Einstellungen in eine Auftragsdatei schreibt. Es wird keine Installation ausgeführt und das Installationsprogramm wird nach der Benutzerinteraktion beendet. Damit kann das Installationsprogramm als eine Art Wizard zum Erstellen der Auftragsdatei für den Netzinstallationsmodus verwendet werden.

"/Path <Pfad der Auftragsdatei>" gibt den Pfad an, in dem die Auftragsdatei zu finden ist, das heißt von wo gelesen (Netzinstallationsmodus) oder wohin (Administratormodus) geschrieben werden soll. Als Standardeinstellung wird die Auftragsdatei im dem Verzeichnis gesucht/erstellt, in dem das Installationsprogramm gestartet wurde.

"/Debug" startet das Installationsmodul im Debugmodus (Kann immer mit angegeben werden) und schreibt eine Textdatei mit dem Namen <Modulname>.txt in das Verzeichnis <Windir>Debug. Außerdem wird auf jeden Fall ein Resultfile in das selbe Verzeichnis ausgegeben.

Möglichkeiten der Installation:

Die folgende Tabelle beschreibt, welche Kommandozeilenparameter miteinander in Kombination auftreten dürfen.

	/S	/Install	/Remove	/Upgrade	/Launcher	/Admin	/Path=
Mit UI	-	-	-	-	-	-	-
/S	x	x/-/-	-/x/-	-/-/x	o	o	o
/Install	x	x	-	-	o	o	o
/Remove	x	-	x	-	o	o	o
/Upgrade	x	-	-	x	o	o	o
/Launcher	x	x/-/-	-/x/-	-/-/x	X	-	o
/Admin	-	-	-	-	-	x	o
/Path=	x/x/x/-	x/-/-/-	-/x/-/-	-/-/x/-	o/o/o/-	-/-/-/x	X

Die Tabelle wird zeilenweise betrachtet, es müssen alle Bedingungen erfüllt sein. Die linke Spalte entspricht der Ausgangsposition, jede andere Spalte entspricht der zusätzlichen Auswahl. x bedeutet die Parameter müssen gemeinsam auftreten, – bedeutet die Parameter dürfen nicht gemeinsam auftreten, o bedeutet der Parameter ist optional, das heißt er kann in der Kombination auftreten, muß aber nicht.

Sonderfall

- /Install, /Remove und /Upgrade müssen mit /S auftreten, dürfen aber nicht gemeinsam auftreten und /S darf auch nicht ohne einen weiteren Parameter sein.
- Wird /Remove angegeben, dann wird keine Steuerdatei benötigt.

Beispiel

- /S (Zeile) braucht entweder /Install, /Remove oder /Upgrade (Spalte)
- /Install (Zeile) braucht /S (Spalte) aber auf keinen Fall /Remove oder /Upgrade(Spalte).
- /S (Zeile) kann /Path = (Spalte) optional haben
- /Path = (Zeile) in der Kommandozeile, dann muß /S (Spalte) und entweder /Install (Spalte), /Remove (Spalte) oder /Upgrade ebenfalls vorhanden sein und kein /Admin (Spalte) oder kein /S (Spalte), kein /Install (Spalte), kein /Remove (Spalte), kein /Upgrade, kein /Launcher (Spalte), aber /Admin (Spalte).
- /Path ist optional, weil im Standardfall die Auftragsdatei im Installationsverzeichnis gesucht/geschrieben wird.

Wird bei der Installation ohne Benutzerinteraktion die Auftragsdatei nicht gefunden, dann wird die Installation abgebrochen. Wird kein /S angegeben, wird von einer Installation mit Benutzeroberfläche ausgegangen, das heißt es dürfen keine weiteren Kommandozeilenparameter für Unattended-Installation angegeben sein und es darf auch keine Auftragsdatei im Installationsverzeichnis bzw. in der "/Path-Angabe vorhanden sein. Ein Obermodul ist selbst für die Erkennung eines Upgrades verantwortlich. Untermodule werden mit dem Schalter /Upgrade aufgerufen, wenn nur ein Upgrade durchgeführt werden soll.

Als Ausführungsbeispiel für das erfindungsgemäße Setup-Verfahren wird im folgenden ein Modulgerüst für Installationsmodule mit dem Wise-Installer nach den oben beschriebenen Ablaufplänen beschrieben. Dieses Modulgerüst deckt alle von einem Installationsmodul auszuführenden Aufgaben ab. Damit ist nur noch die Implementierung der modulspezifischen Bereiche notwendig. Das Modulgerüst besteht aus verschiedenen Wise-Dateien, die in einer Include-Hierarchie verschachtelt sind. Das Modulgerüst gilt sowohl für das Installations- als auch für das Deinstallationsprogramm. Verschiedene Wise-Dateien werden in beiden Teilen des Modulgerüstes benötigt.

Die Wise-Dateien, die mit MS enden sind modulspezifisch und müssen vom Entwickler des Installationsprogrammes (Modul) angepaßt werden. Alle anderen Dateien sind Dateien des Modulgerüstes und dürfen nur nach reiflicher Überlegung, bei auftretenden Fehlern oder bei neuen Erweiterungen geändert werden. Die Dateien des Modulgerüstes enthalten die gesamte Logik der Ablaufpläne. Das Modulgerüst muß immer so allgemein gehalten werden, daß sich damit alle Arten von Modulen (Obermodule, Untermodule) erstellen lassen.

Für die Codierung in Wise ist zu beachten, daß innerhalb des Modulgerüstes eine Vielzahl von Variablen für die interne Logik verwendet wird, die unter Umständen nicht überschrieben werden dürfen. Es muß unbedingt darauf geachtet werden, daß auf Variablen, auf die nur lesend zugegriffen werden darf wirklich nicht schreibend zugegriffen wird, weil sonst im Modulgerüst unvorhersehbare Fehler auftreten können.

Die **Fig.** 6 und 7 zeigen beispielhaft ein Modulgerüst für eine Wise-Installation bzw. eine Deinstallation. Im folgenden werden die hierbei verwendeten Dateien kurz beschrieben.

Die Datei DVSetup.wse ist das Hauptskript für das Installationsprogramm, von dem aus alle anderen Wise-Dateien included werden. Über diese Datei werden die Eigenschaften des Installationsprogrammes eingestellt, die dann für alle Module gelten. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

In der Datei Init_MS.wse werden alle modulspezifischen Initialisierungen der Variablen durchgeführt. Ein Teil dieser Variablen steuert den weiteren Ablauf und die Funktionalität des Installationsmoduls. Diese Datei ist modulspezifisch und muß angepaßt werden.

Die Datei DVInitialize.wse bereitet die Installation vor, und innerhalb diese Skriptes werden wichtige Variablen initialisiert. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

Die Datei CheckCMDLINE.wse überprüft, ob die Kommandozeile die richtige Kombination der Kommandozeilenparameter enthält. Stimmt der Aufruf (Schnittstelle) des Installationsmoduls nicht, wird die Installation beendet. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

Die Datei CheckMAINDIR.wse überprüft, ob bereits ein anderes Produkt der Produktfamilie (z. B. DeskView) installiert ist und in welches Verzeichnis diese installiert wurde. Wurde ein anderes Produkt gefunden, sollte ebenfalls in dieses Verzeichnis installiert werden. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

Die Datei M_Premium.wse ermittelt, welches Motherboard im PC eingebaut ist (Main/Premium) und ob ein Lizenzschlüssel für die Installation eingegeben werden muß oder nicht.

In der Datei CheckPC_MS.wse können verschiedene Aktionen zwischen dem Initialisieren der Variablen und dem Anzeigen des ersten Dialogs bzw. vor dem Lesen der Auftragsdatei ausgeführt werden. Üblicherweise wird diese Datei dazu verwendet, die notwendigen Hardware- bzw. Softwarevoraussetzungen auf dem Zielsystem zu überprüfen (z. B. Filter für die Betriebssystemversion auf der installiert werden darf, Abprüfen auf den richtigen PC-Typ, etc. Zur Vereinfachung können die allgemeine Datei CheckPC.wse oder andere Wise-Dateien mit fest vorgegebener Funktionalität included werden. Diese Datei ist modulspezifisch und muß angepaßt werden.

Die Datei CheckPC.wse überprüft, ob die Installationsbedingungen auf dem PC erfüllt sind (z. B. Administratorrechte, usw.). Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

In der Datei InstDialog_MS.wse wird die Benutzeroberfläche der Installation implementiert. Wird die Installation ohne Benutzerinteraktion aufgerufen, dann wird dieses Skript nicht abgearbeitet. Diese Datei ist modulspezifisch und muß angepaßt werden.

In der Datei CheckKEYPreD.wse wird abgeprüft, ob bereits ein Lizenzschlüssel in der Registry für die aktuell zu installierende Applikation eingetragen ist und ob dieser noch gültig ist. Falls bereits ein gültiger Schlüssel vorhanden ist, kann dieser im Key-Dialog angezeigt werden.

In der Datei CheckKEY.wse wird abgeprüft, ob der eingegebene Lizenzschlüssel gültig ist. Wurde ein gültiger Schlüssel eingegeben, wird normal installiert, wurde ein falscher Schlüssel eingegeben, dann besteht die Möglichkeit entweder eine Standardversion oder eine Trialversion zu installieren, falls das Setupmodul dies unterstützt. Wird die Installation ohne Benutzerinteraktion ausgeführt, dann wird der Lizenzschlüssel aus der Auftragsdatei gelesen.

Die Datei InstSilent.wse ist zusammen mit der Datei InstSilent_MS.wse das Gegenstück zu InstDialog_MS.wse. Wird

die Installation ohne Benutzerinteraktion aufgerufen, dann wird diese Skript abgearbeitet. Was bei der Installation mit Benutzeroberfläche vom Benutzer eingegeben wird, wird teilweise in dieser Datei aus der Auftragsdatei gelesen. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

Die Datei InstSilent_MS.wse ist die modulspezifische Erweiterung zu InstSilent.wse. Wird die Installation ohne Benutzerinteraktion aufgerufen, dann wird dieses Skript abgearbeitet. Was bei der Installation mit Benutzeroberfläche vom Benutzer eingegeben wird, sollte in dieser Datei aus der Auftragsdatei gelesen werden, wenn es nicht bereits durch InstSilent.wse erledigt wird. Zudem können hier noch spezielle Aktionen ausgeführt werden. Diese Datei ist modulspezifisch und muß angepaßt werden.

In der Datei DVModuls.wse werden alle Untermodule aufgerufen und installiert, die über die Moduldefinitionsdatei angegeben werden. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

In der Datei InstModuls_MS.wse werden alle Untermodule angegeben, die nicht über die Moduldefinitionsdatei installiert werden können und immer installiert werden sollen. Sollen keine Untermodule installiert werden, so muß diese Datei leer sein. Diese Datei ist modulspezifisch und muß angepaßt werden.

Das Skript CheckModuls.wse überprüft ob ein angegebenes Modul vorhanden ist und installiert werden kann. Wurde das Modul nicht gefunden, so wird die Installation abgebrochen und das System wieder zurückgesetzt. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

Die Datei InstModuls.wse ruft alle angegebenen Module als Untermodule auf und installiert diese nach dem Ablaufplan. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

In der Datei Inst_MS.wse wird die eigentliche Installation durchgeführt. Alle Dateien die kopiert werden sollen, alle Registryeinträge die angelegt werden sollen werden hier angegeben. Wird das Installationsskript zu groß, kann es vom Entwickler des Installationsprogrammes in mehrere Include-Skripte unterteilt werden. Diese Datei ist modulspezifisch und muß angepaßt werden.

In der Datei DVFinish.wse werden die abschließenden Aufgaben der Installation erledigt. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

Die Datei UnInst.wse ist das Hauptskript für das Deinstallationsprogramm, von dem aus alle anderen Wise-Dateien included werden. Über diese Datei werden die Eigenschaften des Deinstallationsprogrammes eingestellt, die dann für alle Module gelten. Diese Datei gehört zum Modulgerüst und darf nicht verändert werden.

In der Datei UnInstModuls_MS.wse werden die installierten Deinstallationsprogramme der Untermodule zur Deinstallation aufgerufen. Diese Datei ist modulspezifisch und muß angepaßt werden.

In der Datei UnInstDel_MS.wse wird die eigentliche Deinstallation durchgeführt, d. h. hier wird z. B. die Install.log aufgerufen und es wird alles gelöscht, was nicht über die Install.log gelöscht werden kann. Diese Datei ist modulspezifisch und muß angepaßt werden.

Patentansprüche

1. Setup-Verfahren zum Installieren und Deinstallieren von Software-Produkten oder -produktfamilien, die wenigstens eine Komponente gemeinsam nutzen, wobei die gemeinsame Komponente bei der Installation der Produkte installiert oder einem Update unterworfen und bei der Deinstallation deinstalliert wird, wenn sie nicht mehr benötigt wird, **dadurch gekennzeichnet**, daß das Setup-Verfahren durch Module ausgeführt wird, die nacheinander aufgerufen und abgearbeitet werden, und daß die gemeinsamen Komponenten als eigenständige Module innerhalb des Setup-Verfahrens ausgeführt werden und dazu ein eigenes Installationsprogramm und Deinstallationsprogramm ausführen.

2. Verfahren nach Anspruch 1, dadurch gekennzeichnet, daß in der gemeinsamen Komponente festgestellt und gespeichert wird, wie oft sie von anderen Setup-Programmen zur Installation bzw. Deinstallation aufgerufen wurde, und daß die gemeinsame Komponente dann automatisch deinstalliert wird, wenn sie von der letzten Produktinstallation zur Deinstallation aufgerufen wird, die die gemeinsame Komponente noch benutzt hat.

3. Verfahren nach Anspruch 1 oder 2, dadurch gekennzeichnet, daß die Module in eine Hierarchie aus wenigstens einem Obermodul mit Benutzeroberfläche und Untermodulen ohne Benutzeroberfläche eingegliedert werden und daß die Module entsprechend der Hierarchie abgearbeitet werden.

4. Verfahren nach Anspruch 3, dadurch gekennzeichnet, daß beim Hinzufügen eines neuen Obermoduls ein bisheriges Obermodul als Untermodul aufgerufen wird.

5. Verfahren nach Anspruch 4, dadurch gekennzeichnet, daß das neue Obermodul die gesamte Benutzeroberfläche beinhaltet.

6. Verfahren nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß die Hierarchie bzw. die Aufrufreihenfolge in einer Anweisungstabelle festgehalten wird.

7. Verfahren nach Anspruch 6, dadurch gekennzeichnet, daß die Anweisungstabelle außerhalb der Module angelegt wird.

8. Verfahren nach Anspruch 1 oder 2, dadurch gekennzeichnet, daß beim Start einer Installation geprüft wird, ob ein Zählerstand in einem Modulcounter vorhanden ist, daß, wenn kein Zählerstand im Modulcounter vorhanden ist, und es sich damit um eine Neuinstallation handelt, der Zählerstand auf 1 gesetzt, ein Uninstallstring gesetzt und ein Displayname erzeugt wird, daß, wenn ein Zählerstand im Modulcounter vorhanden ist, ein Uninstallstring erzeugt und der Zählerstand des Modulcounters erhöht wird, wenn das Modul zum ersten Mal als Obermodul aufgerufen wurde, und daß dann die Installation der gemeinsamen Komponente durchgeführt wird.

9. Verfahren nach einem der Ansprüche 1, 2 oder 8, dadurch gekennzeichnet, daß, wenn nach der Installation der gemeinsamen Komponente eine weitere gemeinsame Komponente in Form eines Untermoduls installiert wird, geprüft wird, ob der Untermodul bereits installiert war, daß, wenn der Untermodul bereits installiert war und kein Update vorliegt, der Zählerstand des Modulcounters des Untermoduls erhöht, der Untermodul als Client eingetragen, die Deinstallationsroutine vermerkt und die Installation des Untermoduls durchgeführt wird, und daß, wenn der Un-

termodul noch nicht installiert war, der Zählerstand des Modulcounters des Untermoduls um den Wert des eigenen Modulcounters erhöht, das Update-Flag gelöscht, der Untermodul als Client eingetragen, die Deinstallationsroutine vermerkt und die Installation des Untermoduls durchgeführt wird.

10. Verfahren nach Anspruch 1 oder 2, dadurch gekennzeichnet, daß bei der Deinstallation geprüft wird, ob ein anderer Modul aufgerufen werden soll, daß, wenn ja, die Module zur Deinstallation aufgerufen werden, daß, wenn im Client noch ein Zählerstand im Modulcounter vorhanden ist, mit der Deinstallation fortgefahren und u. U. ein weiterer Modul aufgerufen wird, und daß, wenn im Client kein Zählerstand mehr im Modulcounter vorhanden ist, der Clientvermerk gelöscht und die Deinstallationsroutine fortgesetzt wird.

11. Verfahren nach einem der vorhergehenden Ansprüche, dadurch gekennzeichnet, daß, wenn der Zählerstand in dem Modulcounter gleich "1" ist, geprüft wird, ob es einen Zählerstand im dem Sharedcounter gibt, daß, wenn ja, eine Deinstallation durchgeführt wird, die Files gelöscht werden und der Sharedcounter erniedrigt wird, daß, wenn nein, alle Dateien deinstalliert und alle Einträge gelöscht werden, und das Modul deinstalliert wird und daß, wenn der Zählerstand in dem Modulcounter ungleich "1" ist, der Zählerstand im Modulcounter nur um "1" erniedrigt wird.

Hierzu 5 Seite(n) Zeichnungen

- Leerseite -

Nummer: DE 199 24 610 A1
Int. Cl.[7]: G 06 F 9/445
Offenlegungstag: 11. Januar 2001

FIG 1

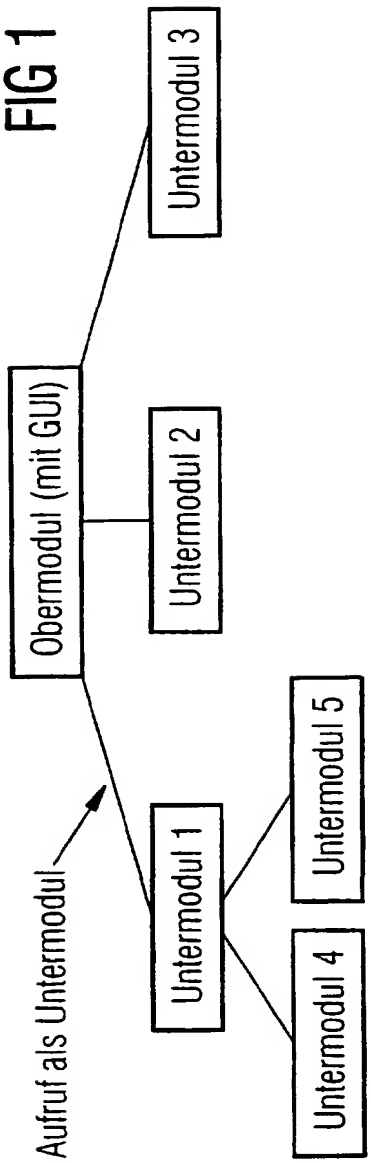
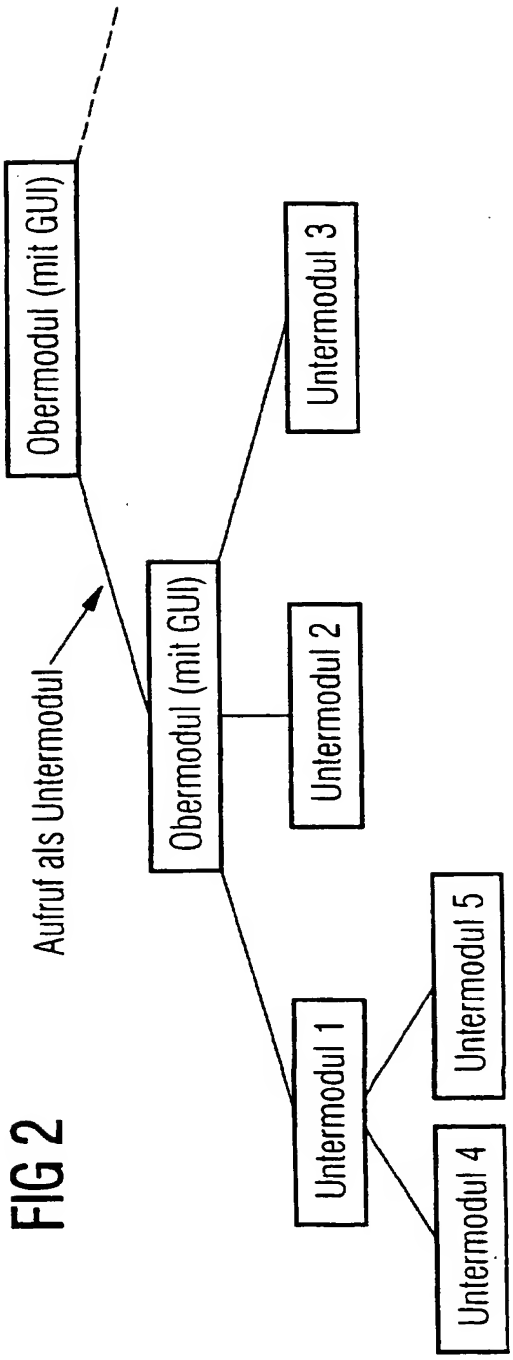
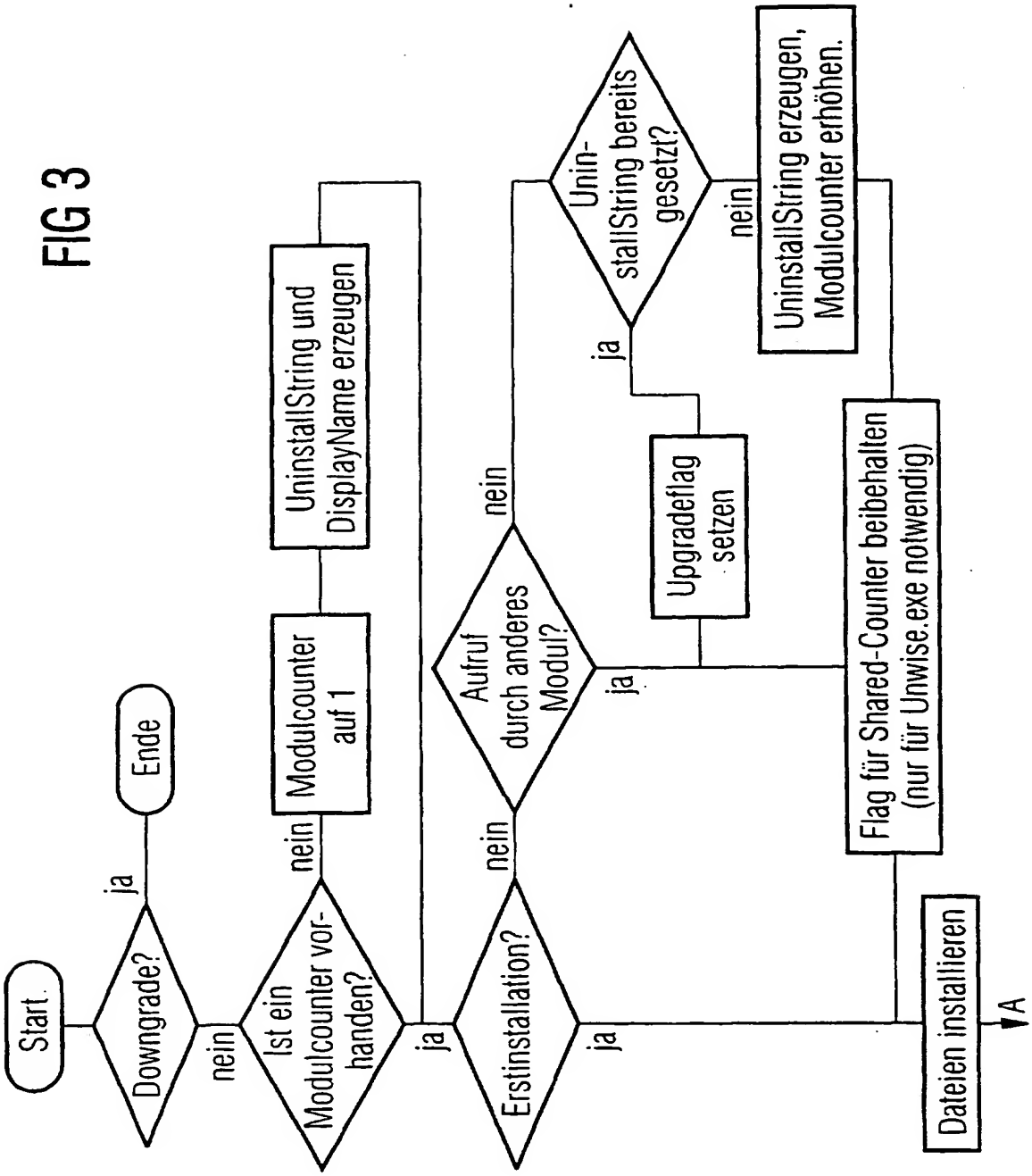


FIG 2



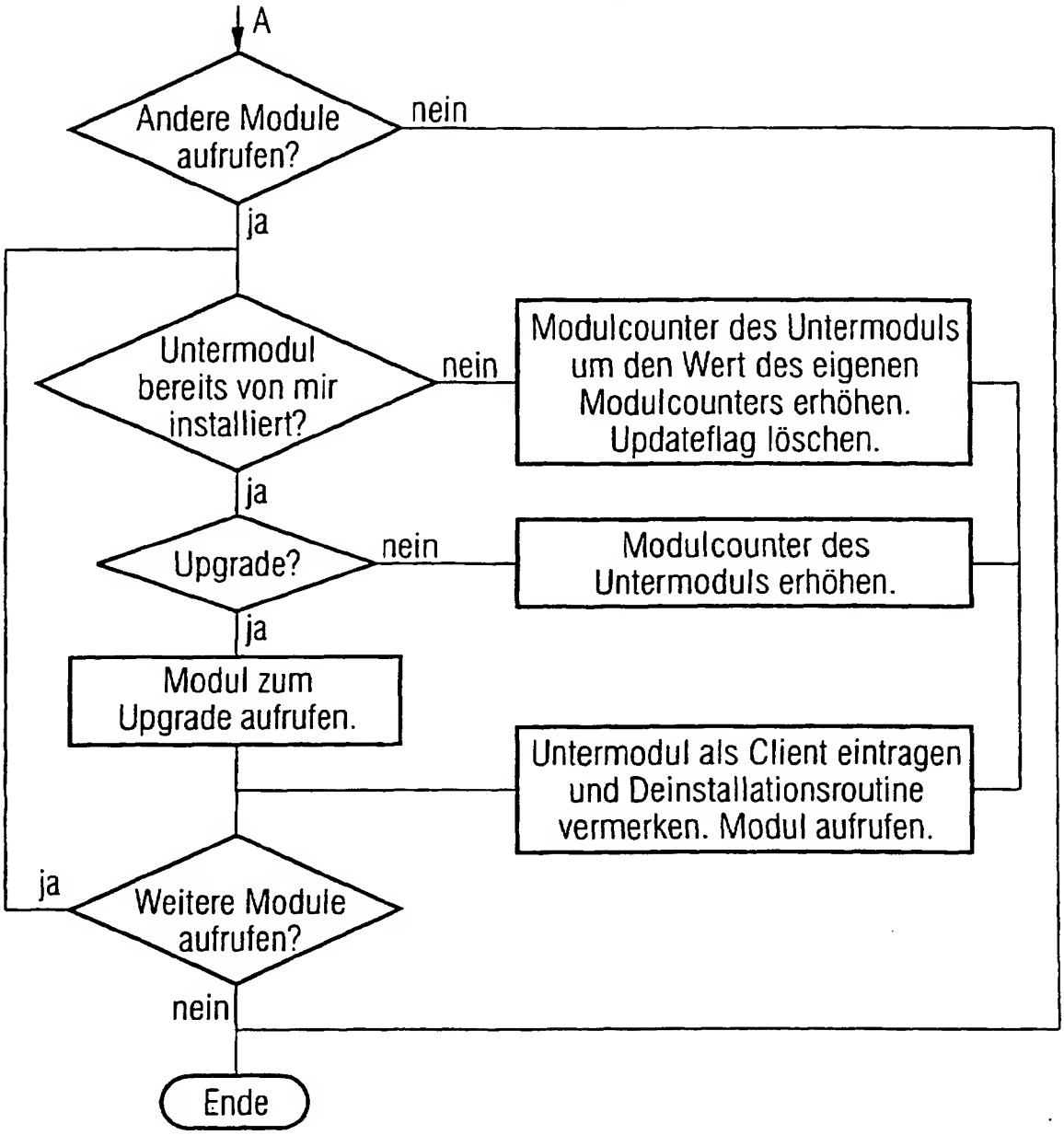
Nummer: DE 199 24 610 A1
Int. Cl.[7]: G 06 F 9/445
Offenlegungstag: 11. Januar 2001

FIG 3



Nummer: DE 199 24 610 A1
Int. Cl.[7]: G 06 F 9/445
Offenlegungstag: 11. Januar 2001

FIG 4



Nummer: DE 199 24 610 A1
Int. Cl.[7]: G 06 F 9/445
Offenlegungstag: 11. Januar 2001

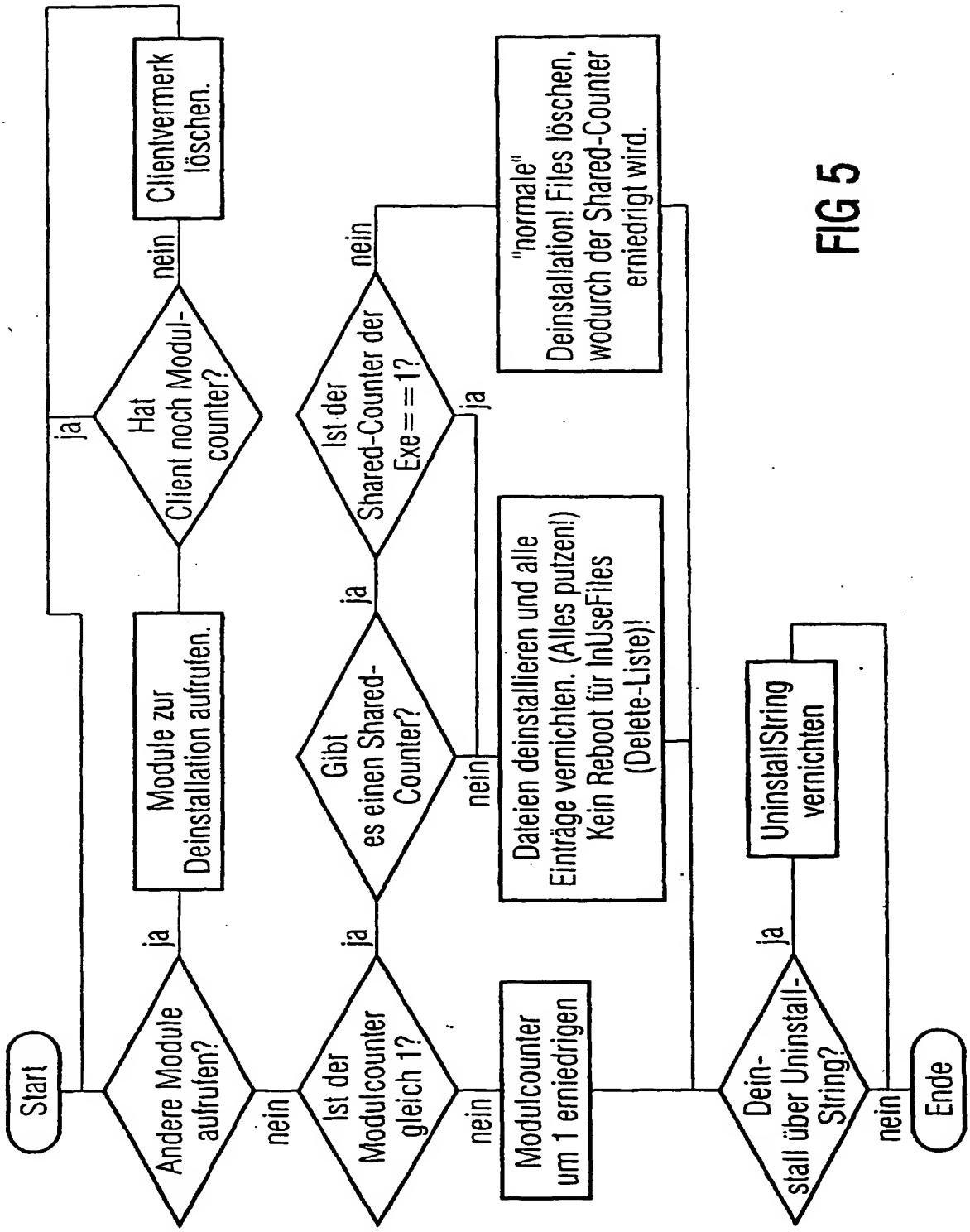


FIG 5

Nummer: DE 199 24 610 A1
Int. Cl.[7]: G 06 F 9/445
Offenlegungstag: 11. Januar 2001

FIG 6

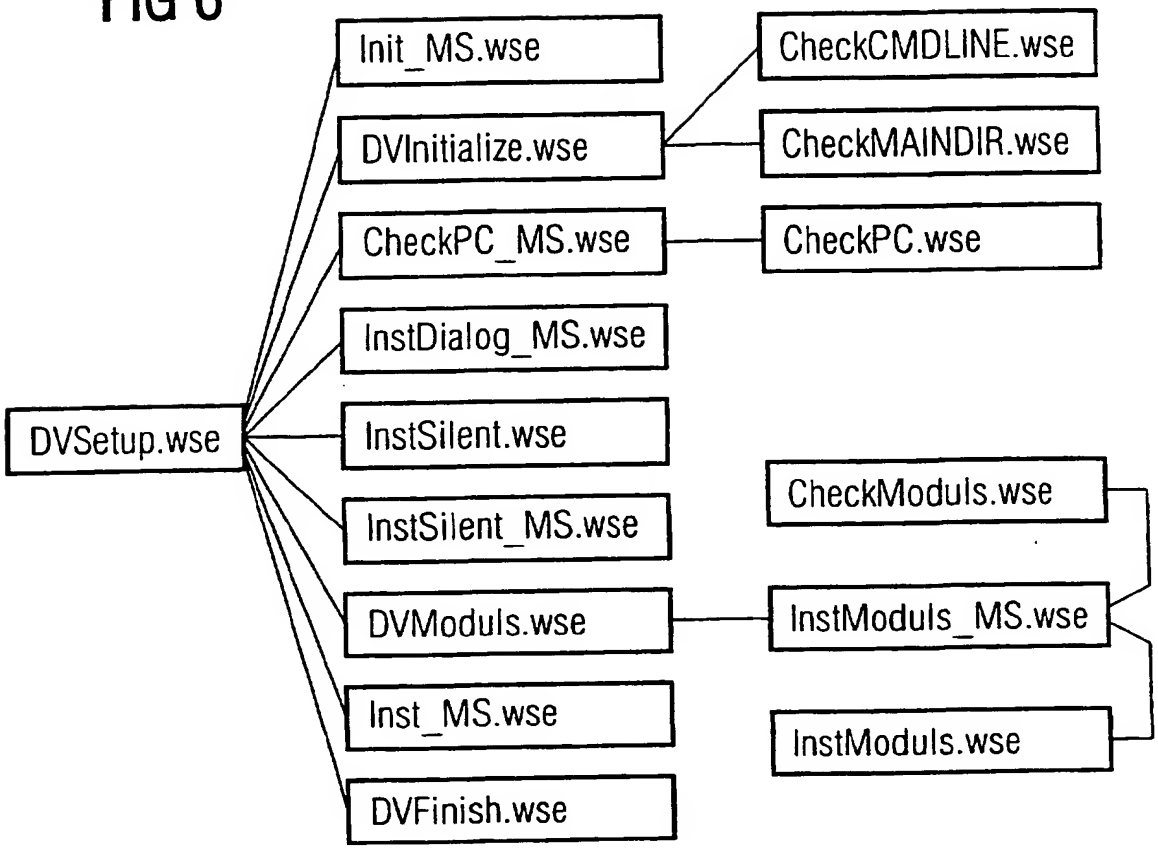


FIG 7